

GPUs: Engines for Future High-Performance Computing

John Owens

**Assistant Professor, Electrical and Computer
Engineering**

**Institute for Data Analysis and Visualization
University of California, Davis**

Report Documentation Page				Form Approved OMB No. 0704-0188	
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE 01 FEB 2005		2. REPORT TYPE N/A		3. DATES COVERED -	
4. TITLE AND SUBTITLE GPUs: Engines for Future High Performance Computing				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Electrical and Computer Engineering, Institute for Data Analysis and Visualization University of California, Davis				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release, distribution unlimited					
13. SUPPLEMENTARY NOTES See also ADM001742, HPEC-7 Volume 1, Proceedings of the Eighth Annual High Performance Embedded Computing (HPEC) Workshops, 28-30 September 2004. , The original document contains color images.					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 31	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

GPUs as Compute Engines

10 years ago:

- Graphics done in software

5 years ago:

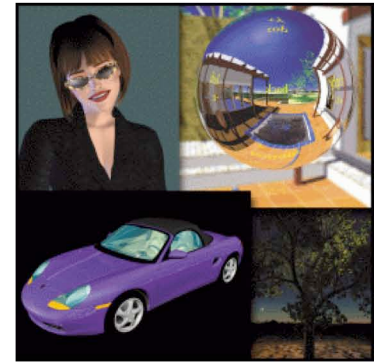
- Full graphics pipeline

Today:

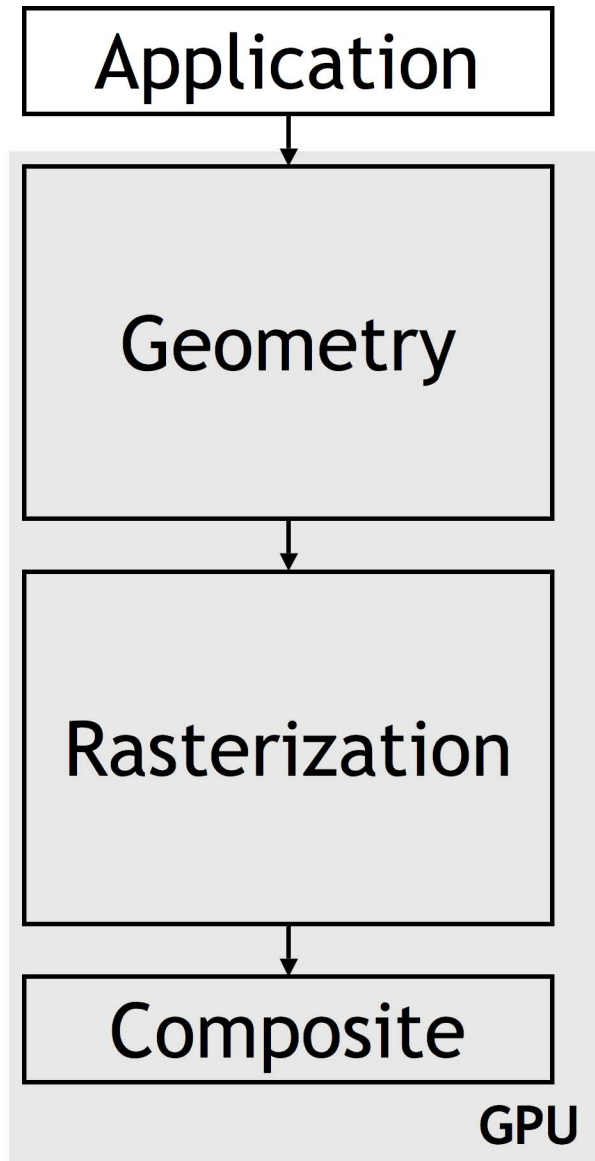
- 40x geometry, 13x fill vs. 5 yrs ago
- Programmable!

Programmable, data parallel
processing on every desktop

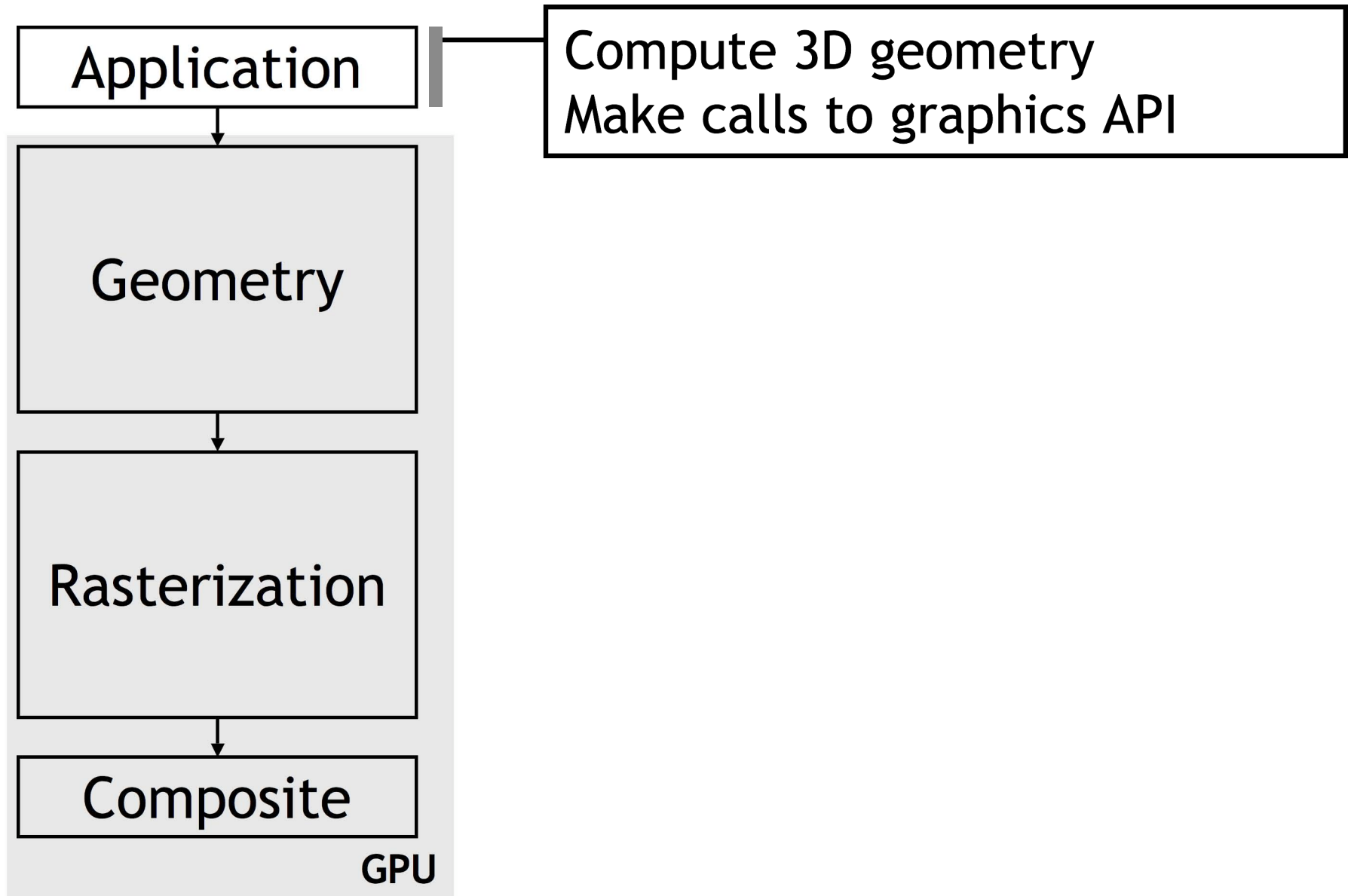
Enormous opportunity to change the
way commodity computing is done!



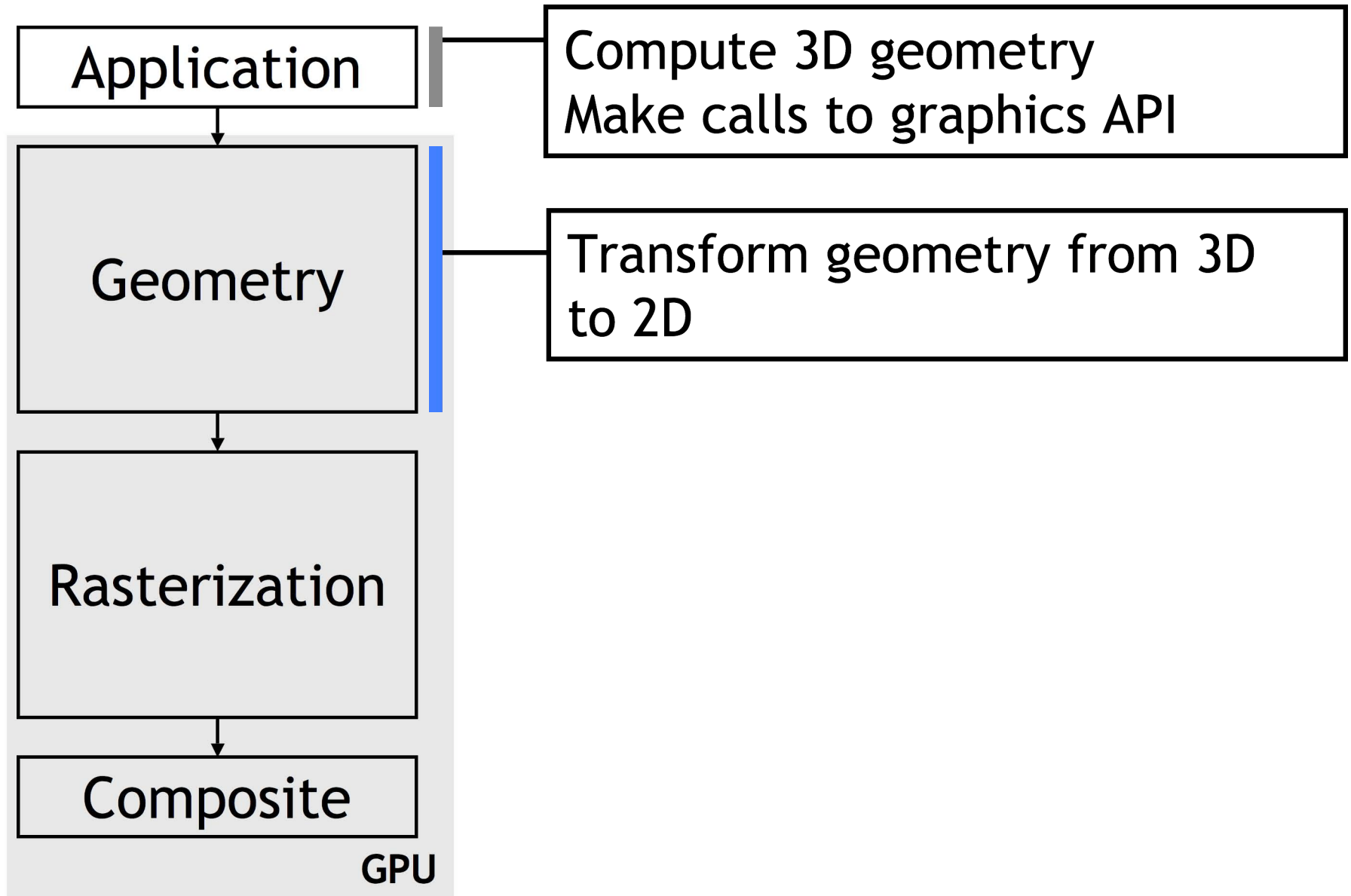
The Rendering Pipeline



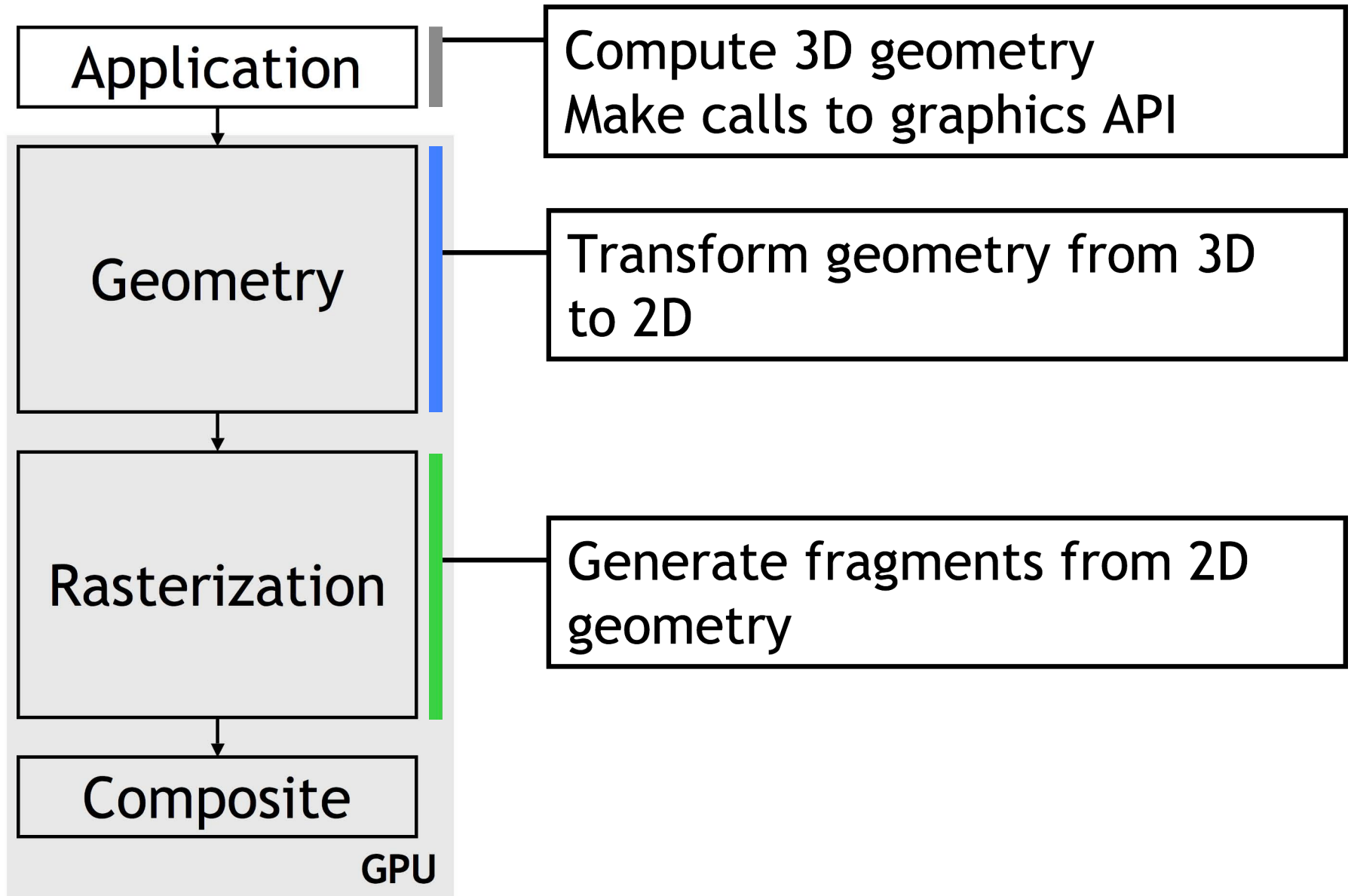
The Rendering Pipeline



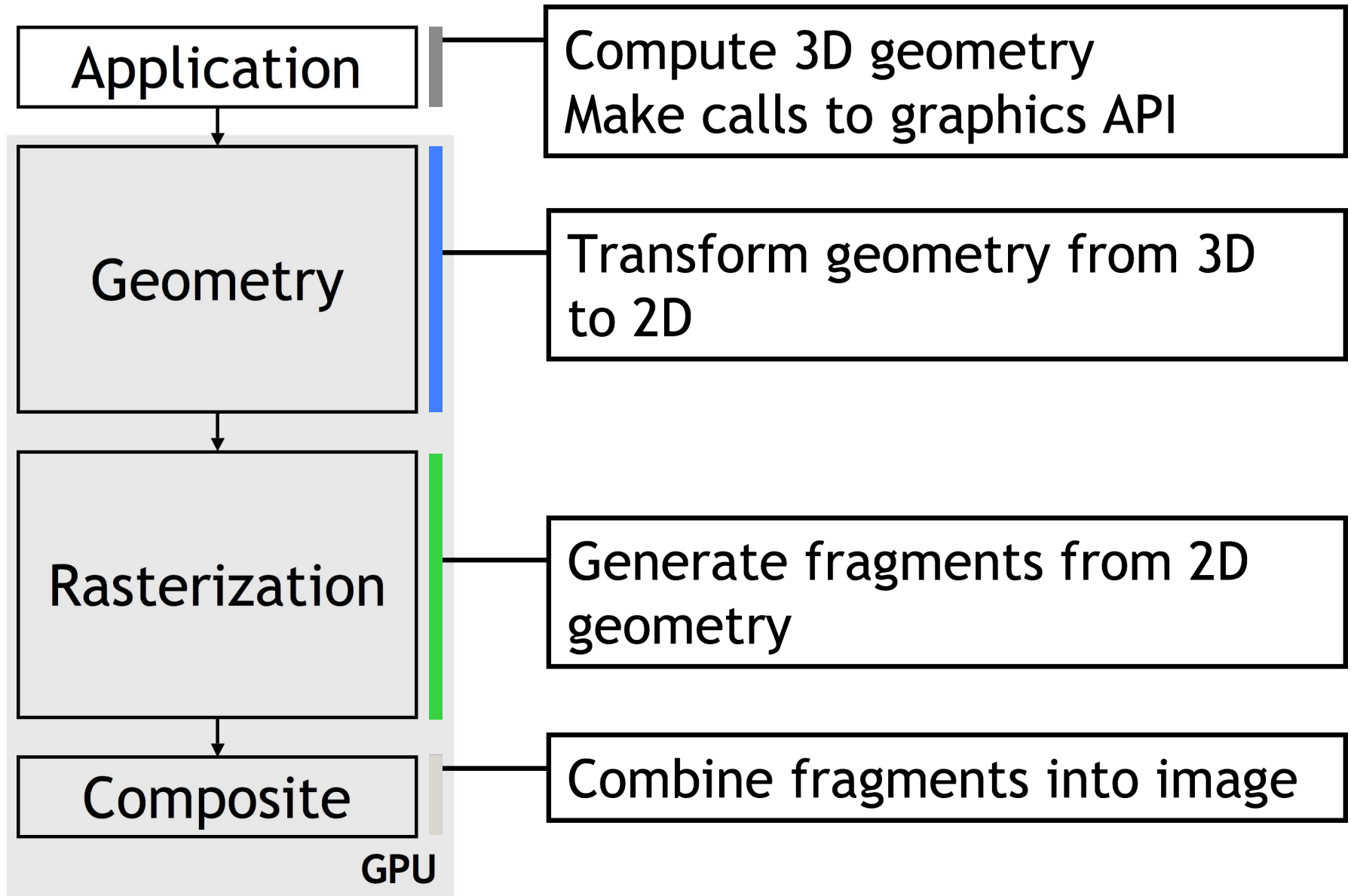
The Rendering Pipeline



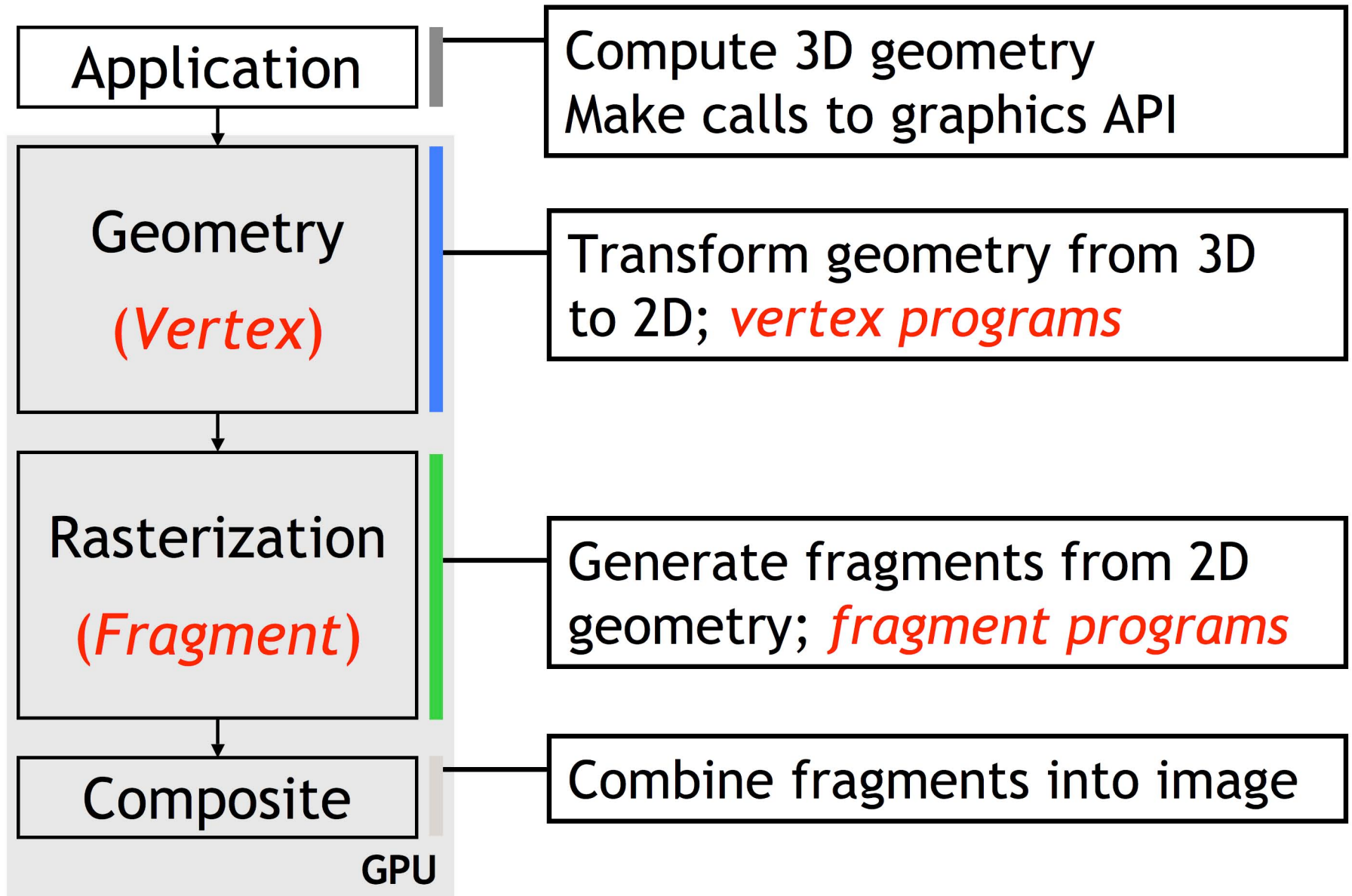
The Rendering Pipeline



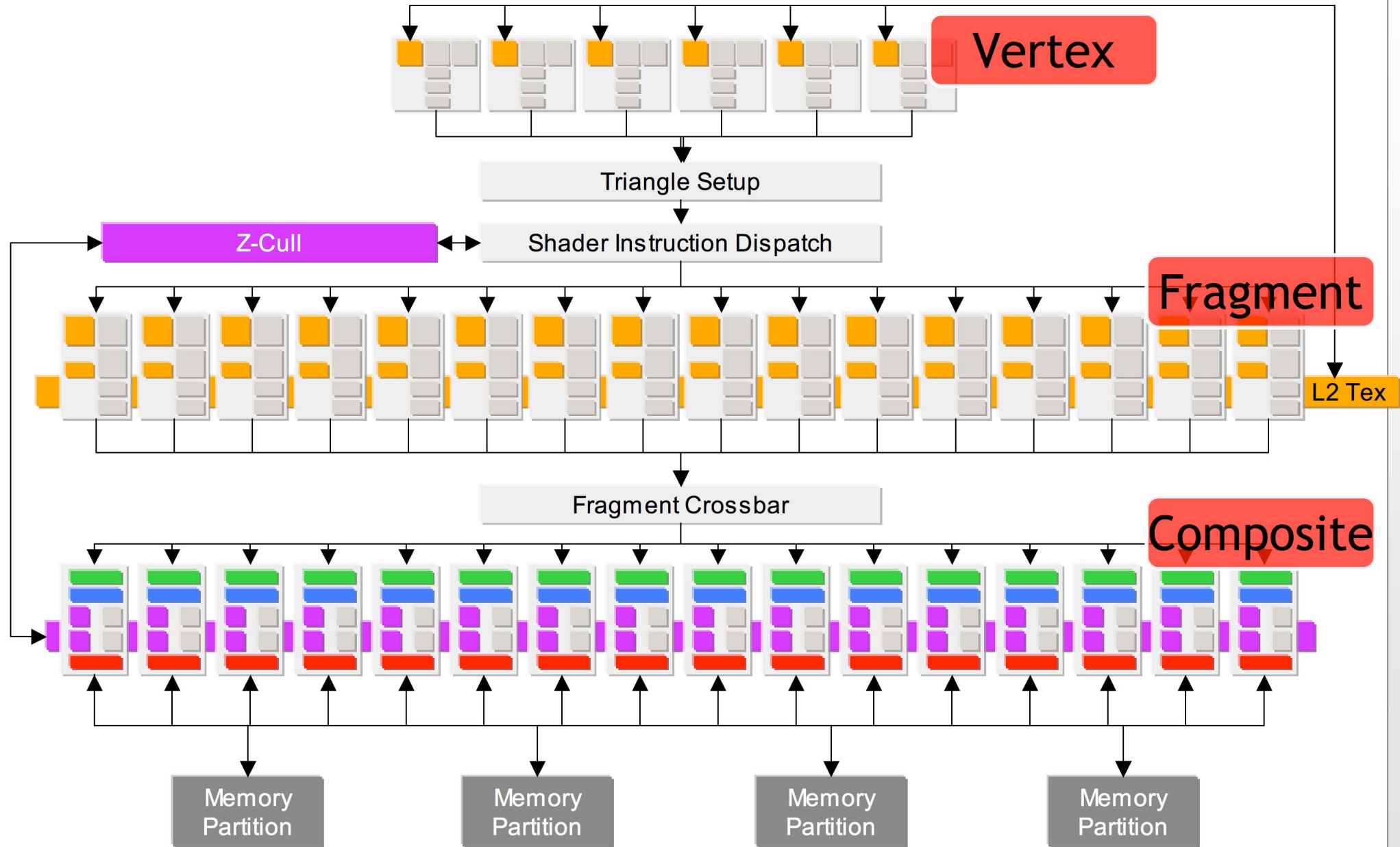
The Rendering Pipeline



The **Programmable** Rendering Pipeline

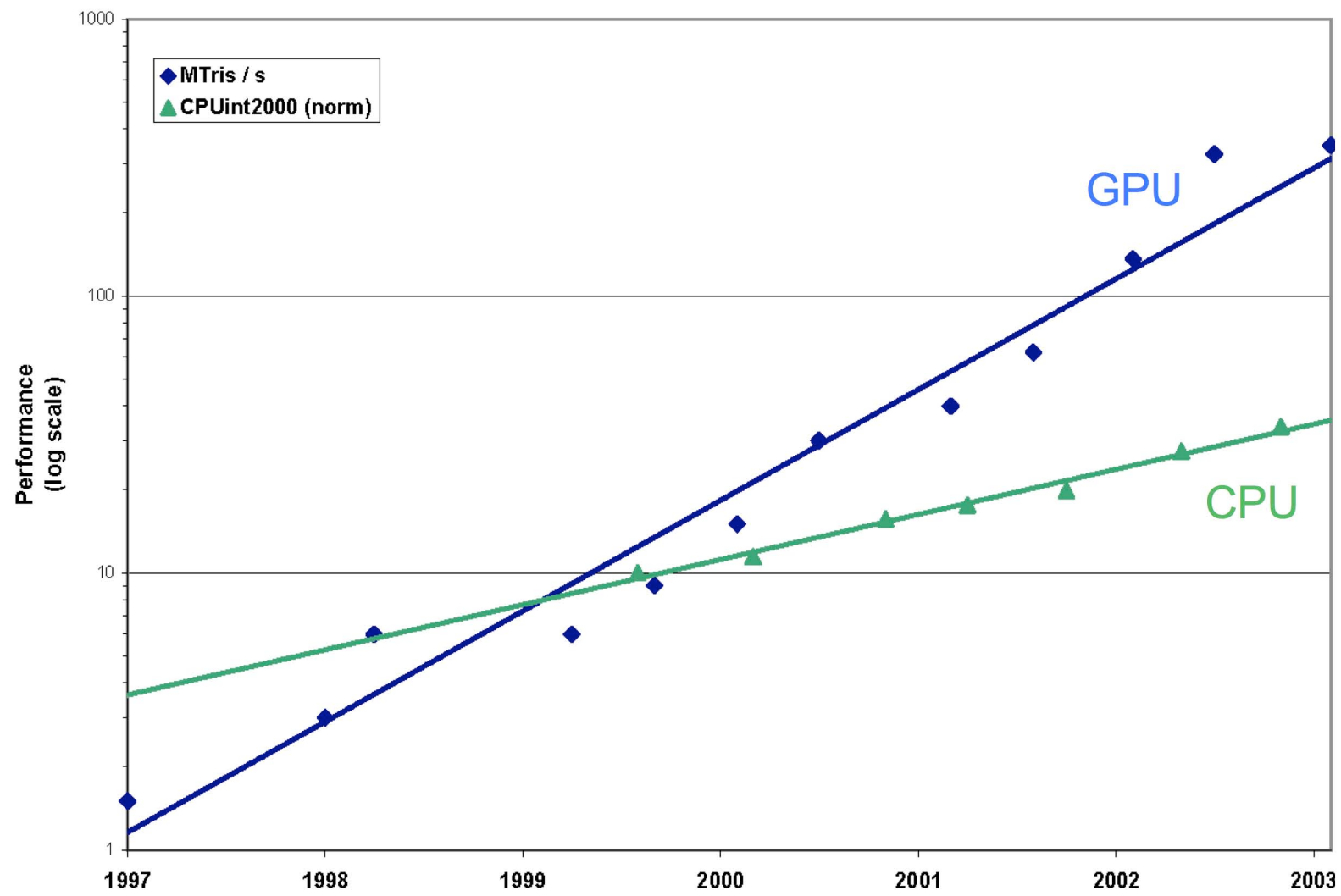


NVIDIA GeForce 6800 3D Pipeline



Courtesy Nick Triantos, NVIDIA

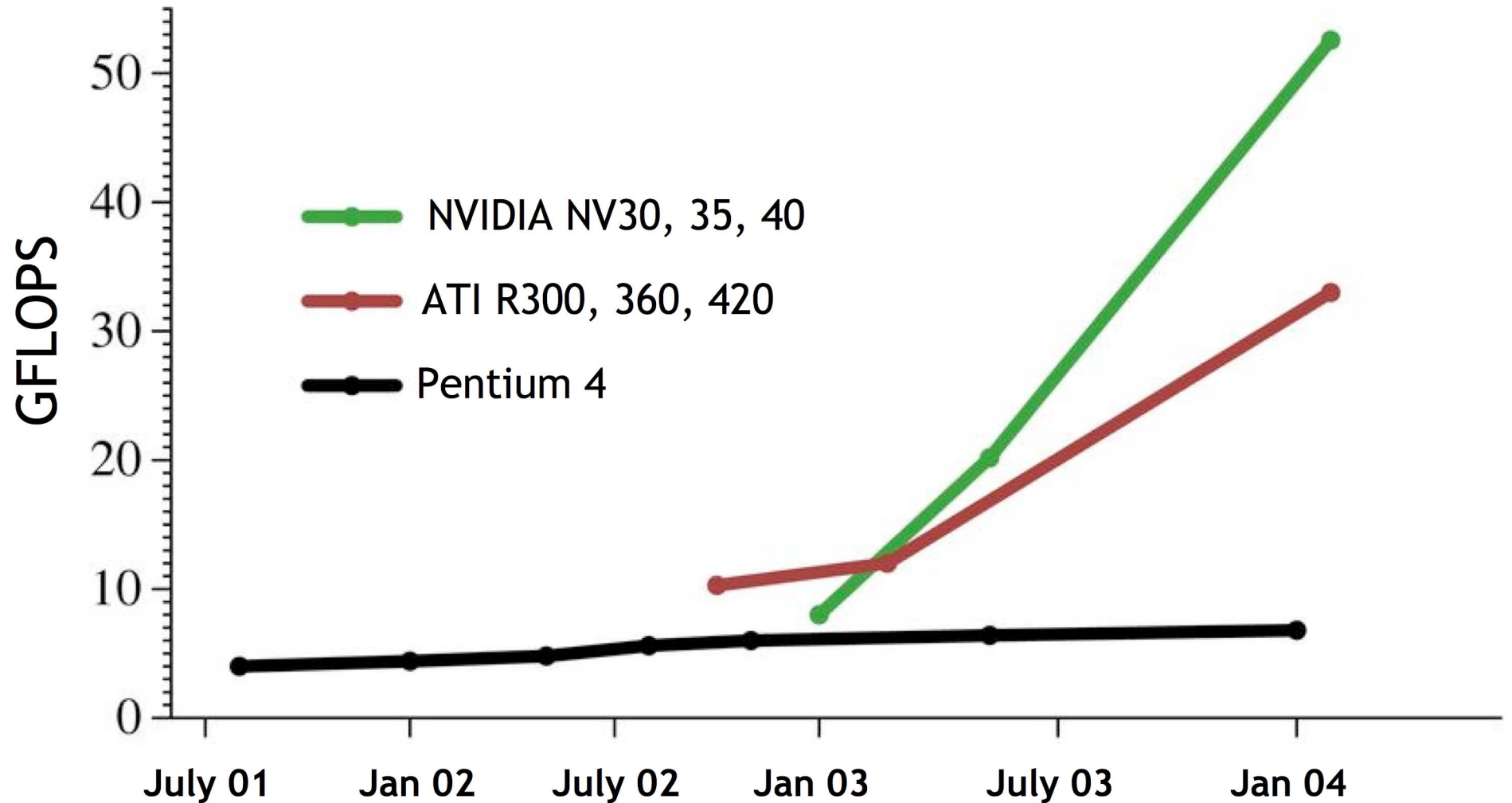
Long-Term Trend: CPU vs. GPU



Courtesy Naga Govindaraju

Recent GPU Performance Trends

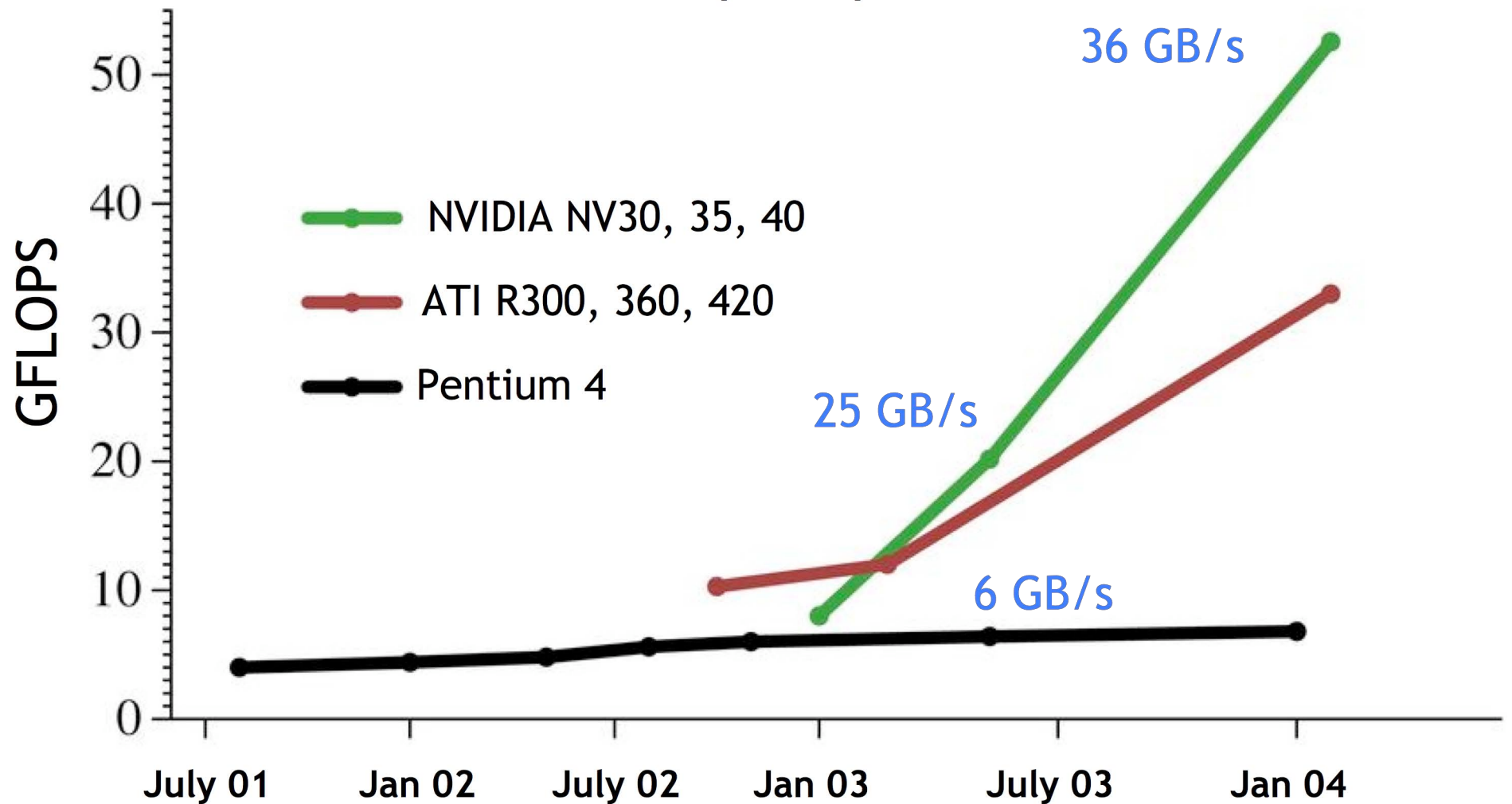
32-bit FP multiplies per second



Courtesy Pat Hanrahan/David Luebke

Recent GPU Performance Trends

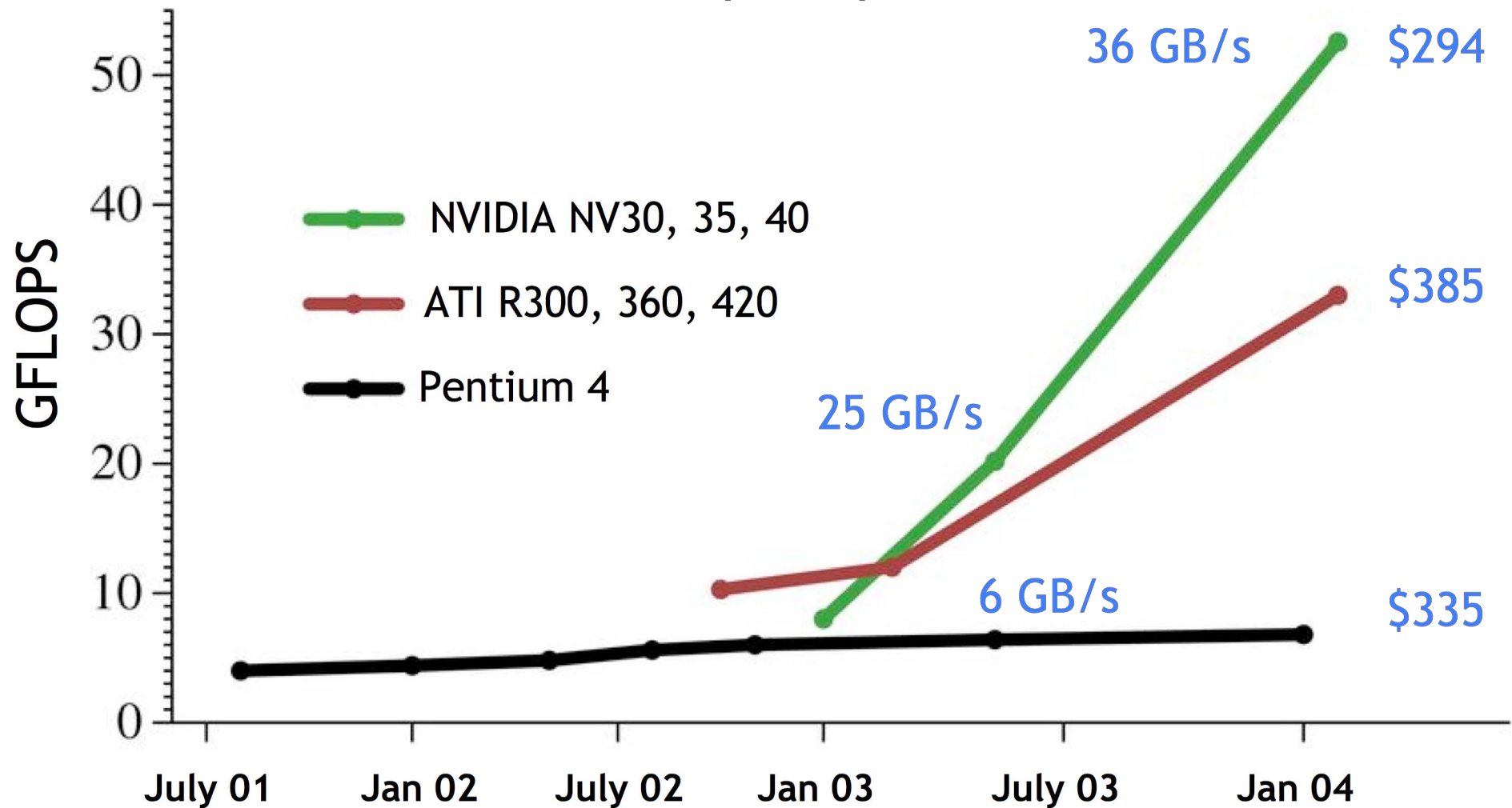
32-bit FP multiplies per second



Courtesy Pat Hanrahan/David Luebke

Recent GPU Performance Trends

32-bit FP multiplies per second



Courtesy Pat Hanrahan/David Luebke

Why Are GPUs Fast?

Characteristics of computation permit efficient hardware implementations

- High amount of parallelism ...
- ... exploited by graphics hardware
- High latency tolerance and feed-forward dataflow ...
- ... allow very deep pipelines
- ... allow optimization for bandwidth not latency

Simple control

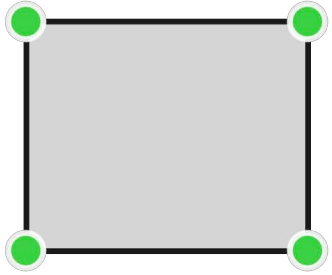
- Restrictive programming model

Competition between vendors

What about programmability? Effect on performance? How hard to program?

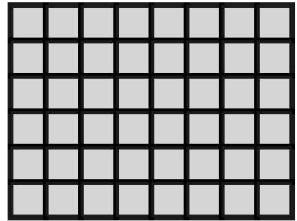
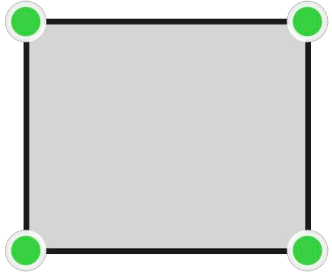
Programming a GPU for GP Programs

Programming a GPU for GP Programs



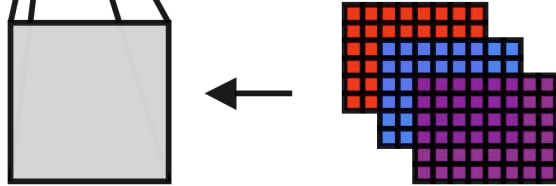
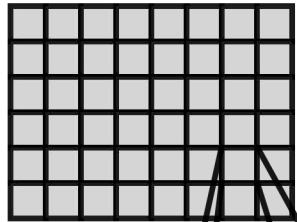
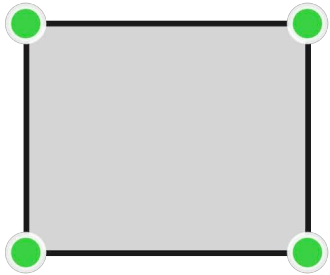
- Draw a screen-sized quad

Programming a GPU for GP Programs



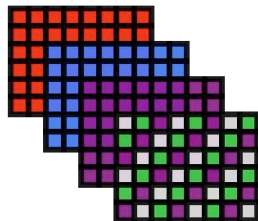
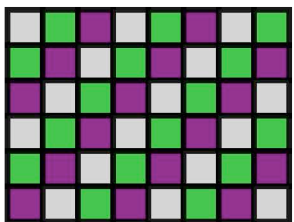
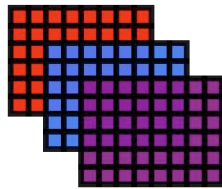
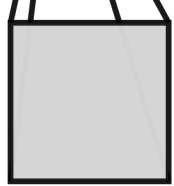
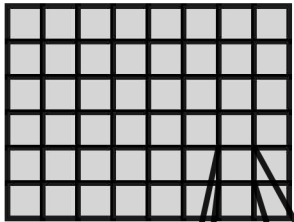
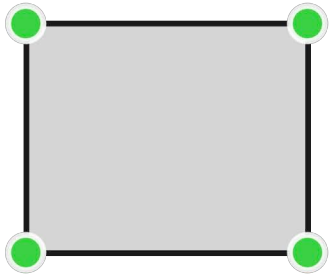
- Draw a screen-sized quad
- Run a SIMD program over each fragment

Programming a GPU for GP Programs



- Draw a screen-sized quad
- Run a SIMD program over each fragment
- “Gather” is permitted from texture memory

Programming a GPU for GP Programs



- Draw a screen-sized quad
- Run a SIMD program over each fragment
- “Gather” is permitted from texture memory
- Resulting buffer can be treated as texture on next pass

GPU Programming is Hard

Must think in graphics metaphors

**Requires parallel programming (CPU-GPU,
task, data, instruction)**

**Restrictive programming models and
instruction sets**

Primitive tools

Rapidly changing interfaces

Challenge: Programming Systems

Programming
Model

High-Level
Abstractions/
Libraries

Low-Level
Languages

Compilers

Performance Analysis Tools

Docs

CPU

Scalar

STL, GNU SL, MPI, ...

C, Fortran, ...

gcc, vendor-specific, ...

gdb, vtune, Purify, ...

Lots

→ *applications*

GPU

Stream?

-

GLSL, Cg, HLSL, ...

Vendor-specific

Shadersmith, NVPerfHUD

None

→ *kernels*

Brook: General-Purpose Streaming Language

Stream programming model

- Treats GPU as streaming coprocessor
- Streams enforce data parallel computing
- Kernels encourage arithmetic intensity
- Streams and kernels explicitly specified

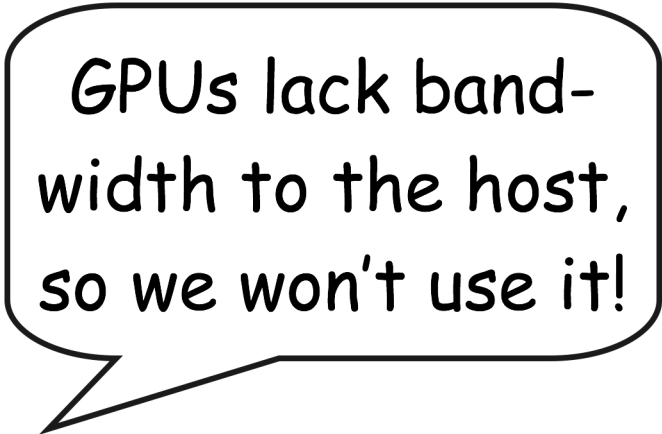
C with stream extensions

Open-source: www.sf.net/projects/brook/

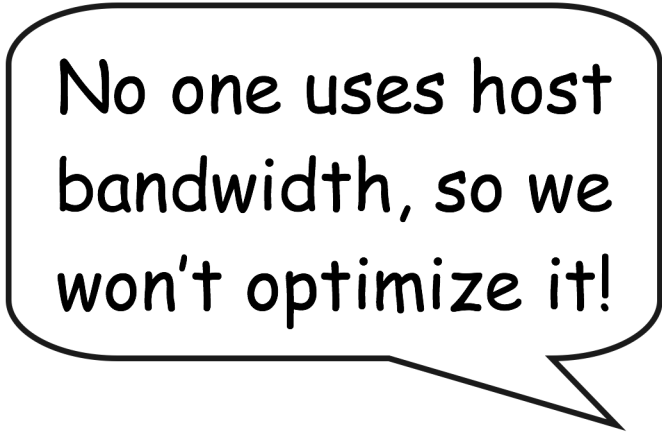
Ian Buck et al., “Brook for GPUs: Stream Computing on Graphics Hardware”,
Siggraph 2004



Challenge: GPU-to-Host Bandwidth



GPUs lack bandwidth to the host, so we won't use it!



No one uses host bandwidth, so we won't optimize it!

Challenge: GPU-to-Host Bandwidth

GPUs lack bandwidth to the host, so we won't use it!

No one uses host bandwidth, so we won't optimize it!

PCI 
EXPRESS®

Challenge: GPU-to-Host Bandwidth

GPUs lack bandwidth to the host, so we won't use it!

No one uses host bandwidth, so we won't optimize it!



- **PCI-E optimizes GPU-to-CPU bandwidth**
 - 16-lane card: 8 GB/s
 - Scalable in future
- **Major vendors support PCI-E cards now**
- **Multiple GPUs supported per CPU - opportunity!**
 - Cheap and upgradable

Challenge: Mobile/embedded market

Why?

- UI, messaging/screen savers, navigation, gaming (location based)

Typical specs (cell-phone class):

- 200-800k gates, ~100 MHz, ~100 mW
- 1-10M vtx/s, 100+M frags/s

What's important?

- Visual quality
- Power-efficient (ops/W)
 - Avoid memory accesses, unified shaders ...
- Low cost



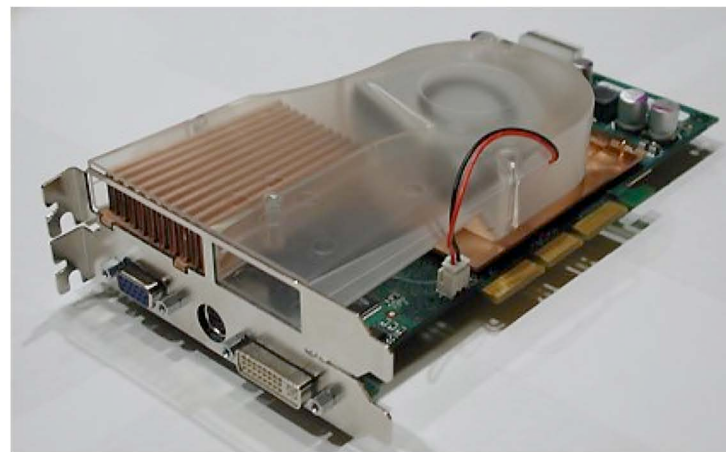
POWERVR
Visionary IP



Challenge: Power

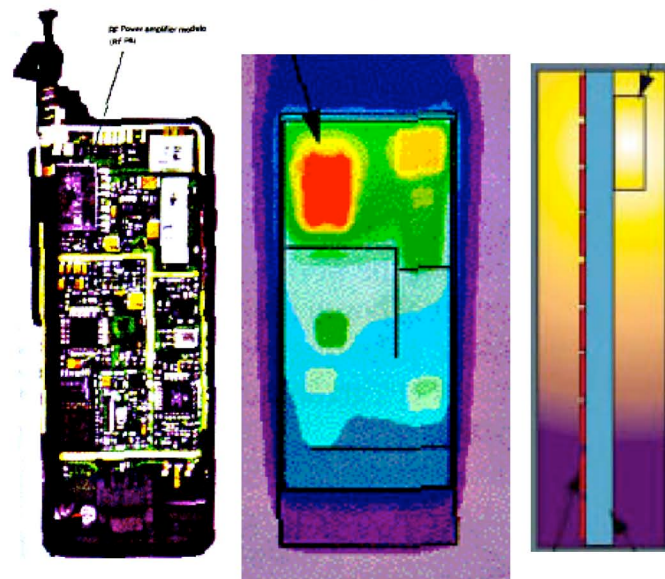
Desktop:

- Double-width cards
- Workstation power supplies; draw power from motherboard



Mobile:

- Batteries improving 5-10% per year
- Ops/W most important



Current GPGPU Research

Image processing [Johnson/Frank/Vaidya, LLNL]

Alternate graphics pipelines [Purcell, Carr, Coombe]

Visual simulation [Harris]

Volume rendering [Kniss, Krüger]

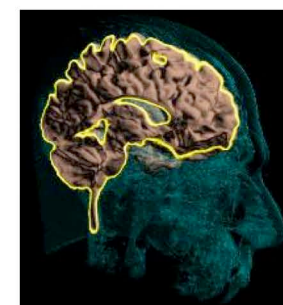
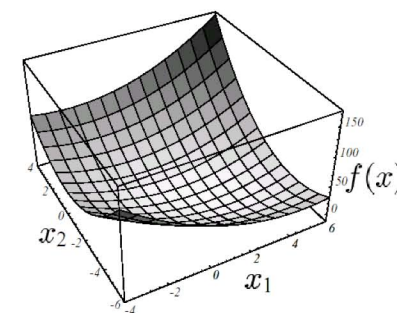
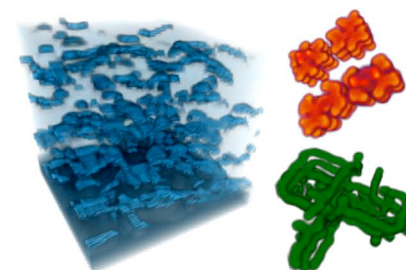
Level set computation [Lefohn, Strzodka]

Numerical methods [Bolz, Krüger, Strzodka]

Molecular dynamics [Buck]

Databases [Sun, Govindaraju]

...



Grand Challenges

Architecture: Increase features and performance without sacrificing core mission

Interfaces: Abstractions, APIs, programming models, languages

- *Many* approaches needed
- Goal: C programs compiling to dynamically-balanced CPU-GPU clusters
- Academic and research community

Applications: Killer app needed!

Acknowledgements

Craig Lund: Mercury Computer Systems

Jeremy Kepner: Lincoln Labs

Nick Triantos, Craig Kolb: NVIDIA

Mark Segal: ATI

Kari Pulli: Nokia

Aaron Lefohn: UC Davis

Ian Buck: Stanford

**Funding: DOE Office of Science, Los Alamos
National Laboratory, ChevronTexaco, UC
MICRO, UC Davis**

For more information ...

GPGPU home: <http://www.gpgpu.org/>

- Mark Harris, UNC/NVIDIA

***GPU Gems* (Addison-Wesley)**



- Vol 1: 2004; Vol 2: 2005

Conferences: Siggraph, Graphics Hardware, GP²

- Course notes: Siggraph '04, IEEE Visualization '04

University research: Caltech, CMU, Duisberg, Illinois, Purdue, Stanford, SUNY Stonybrook, Texas, TU München, Utah, UBC, UC Davis, UNC, Virginia, Waterloo